Video Retrieval by Spatial and Temporal Structure of Trajectories*

James J. Little Zhe Gu

Computer Science Dept.
University of British Columbia
Vancouver, BC, Canada V6T 1Z4
little@cs.ubc.ca

ABSTRACT

Our goal is to enable queries about the motion of objects in a video sequence. Tracking objects in video is a difficult task, involving signal analysis, estimation and often semantic information particular to the targets. That is not our focus - rather, we assume that tracking is done, and turn to the task of representing the motion for query. The position over time of an object results in a motion trajectory, i.e., a sequence of locations. We propose a novel representation of trajectories: we use the path and speed curves as the motion representation. The path curve records the position of the object while the speed curve records the magnitude of its velocity. This separates positional information from temporal information, since position may be more important in specifying a trajectory than the actual velocity of a trajectory. Velocity can be recovered from our representation. We derive a local geometric description of the curves invariant under scaling and rigid motion. We adopt a warping method in matching so that it is robust to variation in feature vectors. We show that R-trees can be used to index the multidimensional features so that search will be efficient and scalable to a large database.

Keywords: video indexing, trajectory, curve representation, motion representation

1. INTRODUCTION

The prevalence of video brings challenges including storage of video on computer systems, real-time synchronized delivery of video, and content-based retrieval. Our work addresses the content-based retrieval task. In a traditional database, text and annotations are used to search for the information. To enhance this approach, content-based query techniques[1,2] use visual features of the images and videos. In image databases (QBIC,[3] Virage[4] and VisualSEEK[5]), users construct queries using graphical interface tools, or provide examples of images, then retrieves the desired images based on the features such as color, texture and shape. Techniques used in image databases can also be utilized in video databases. Frequently video is treated as an extension of images by processing key frames: The video is parsed to obtain individual shots; key frames from each shot are extracted and merged into coherent groups.

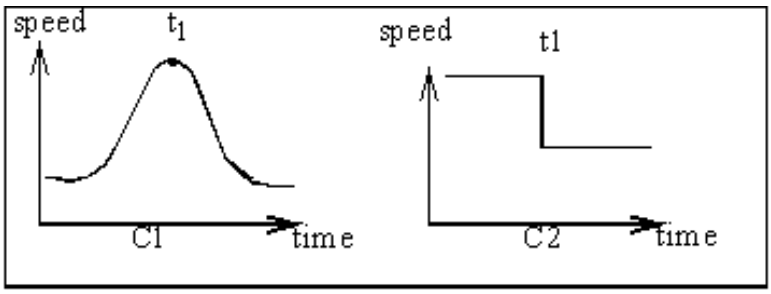


Figure 1. Speed representation, which shows the magnitude of velocity over time.

However, this approach ignores the temporal nature of video. Our work utilizes the motion of objects. The position over time of an object results in a motion trajectory, i.e., a sequence of locations (x_i, y_i), for $i = 1 \ldots n$,

*This research was supported by grants from the Natural Sciences and Engineering Research Council of Canada and the Networks of Centres of Excellence Institute for Robotics and Intelligent Systems.

where n is the number of frames in the sequence. We propose to separate the positional information from the temporal information in a trajectory; we use the *path* and *speed* (the magnitude of velocity) curves as the representation of object motion. Often a user has a strong idea about the path taken by an object in an image. Frequently the temporal characteristics are at best known qualitatively. By separating the two aspects of trajectories we allow the user to select either the path or the speed or both properties of the moving object. Thus we isolate the two elements of a trajectory: the path in the image and the speed at which the body moves along the curve. Given an image sequence, we track a moving object's position and construct its path. We can also compute the speed if the time interval between two frames is known. The speed curve is a 2D curve, where x indicates time and y speed. In Fig. 1, C1 represents the speed of an object that increases until time t1, then decreases. C2 describes an object moving with a constant speed until time t1, then changing to a lower speed.

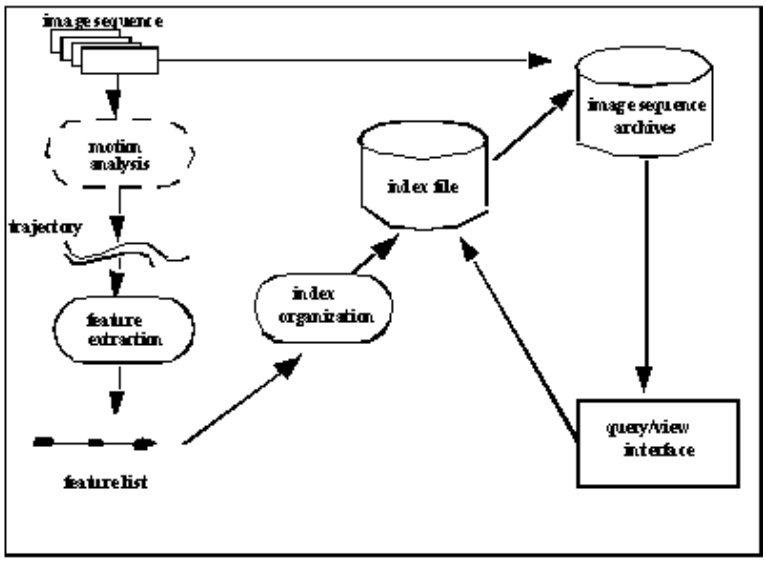


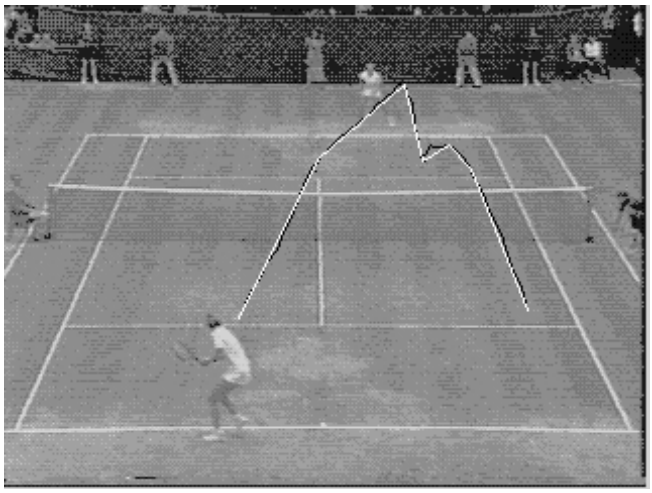
Figure 2. System Architecture

We derive geometric features that are invariant to scaling and rigid motions and preserve local features. The latter is important in searching for subsequences. We adopt a warping method in feature matching so that it will be robust to variation in feature vectors. We use R-trees to index the multidimensional features efficiently; these indexes are scalable to a large database. The system derives the path and speed curve of an image sequence, then extracts the features and inserts them into the database. During retrieval, the system derives features of the query, then searches the indexing tree to retrieve similar ones. The system architecture is shown in Fig. 2. Our approach is useful when a user wants to extract a segment of video containing a certain pattern of motion. For example, a coach may want to search for certain patterns of movement in a tennis match (Fig. 3).

1.1. Overview of existing video indexing approaches

One stream of research attempts to integrate all available media such as audio, video and captions. Chang et al.[4] present a video engine for content-based video retrieval that indexes key frames, image flow, audio and captions. Many techniques apply the methods used in still image databases to the key frames of video sequences. VideoQ[6] includes visual features used in image database as well as a novel interface for QBE. The video is indexed individually by color, texture, shape and motion, as well as temporal features. Several systems use motion as a feature of indexing video.[7–10] Video sequences are processed to extract semantic features. Image sequences are indexed based on the motion properties of objects within the sequence.

Ioka, Dimitrova, Lee and Sahouria index video based on motion of objects mainly, while VideoQ and Virage integrate motion of objects as one of the attributes in the system. Ioka and Kurokawa[7] and Dimitrova and Golshani[11] record the motion of subblocks of the image and use them as features. Lee and Kao[9] presented a qualitative description which enable subsequence queries. Sahouria[8] computes the Haar wavelet transform separately for the X and Y vectors; the first eight coefficients of the projection transform form the search key. VideoQ[6] emphasizes motion as an



(a)

(b)

Figure 3. Tennis: (a) the digitized path (black) and its features (white); (b) the calculated speed curve (black) and its features (white).

attribute; it uses vectors to present the interframe motion, and compare the trails without compressing it. Virage[4] includes motion descriptions (total motion, motion uniformity, panning and tilting) as an indexing feature. Recently Chen and Chang[12] proposed both spatial and spatiotemporal representation of trajectories. Wavelet analysis allows global description of motion and partitions the motion into subtrajectories whose representations are translation invariant.

2. SIMILARITY OF MOTION

The path and speed curves are 2D planar curves, represented by lists of 2D points. The curves representing two trajectories can be compared directly, but the comparison is costly and not invariant to scaling and rigid motions. By feature extraction we simplify the curve representation so that similarity matching will be efficient.

Currently, we derive trajectories manually but they can be derived by tracking or by optical flow.[13] When a user is trying to retrieve videos according to motion information, e.g., find a player skating behind the net in hockey, she may not care about the position of an object in the frame, its size or its orientation. This may not apply in all situations; however, we assume that invariance to translation, rotation and scaling is an objective of our feature extraction method.

2.1. Feature extraction based on curvature

Image sequences are smoothed and then processed to get the path and speed curves. Corners and high curvature points aid object recognition. As well, curvature is invariant to rotation and translation. We use a local approximation to curvature[14]: the distance of a point from the segment connecting its two neighboring points. Points where this measure exceeds a threshold are retained. The threshold must vary with curve size, so we normalize each curve by computing its minimum bounding rectangle. The feature points are fewer than the points in the original curve, but they are still in absolute coordinates and must be abstracted to curve features invariant to scale, translation and rotation.

We derive the maximum curvature feature points and then compute angles between successive segments (Fig. 4) as well as the relative lengths of adjacent segments, i.e., the ratios of the lengths of successive segments, following Stein and Medioni,[15] which has proved a very successful representation for object recognition. The feature values of a curve are: $\{(L_1, A_1), (L_2, A_2), ..., (L_n, A_n)\}$ where L_i indicates the relative length parameter of i^{th} feature value and A_i indicates the angle parameter of i^{th} feature value. n is the length of the feature value linked list; a curve with $n+2$ points has n feature values. Angles are not affected by scaling; lengths are, but not relative lengths. Our representation is unaffected by rotation, translation and scaling. As each pair of index values only depends on three feature points, it preserves the local information of the curve, so this method makes it possible to compare only a subsegment of the curve.

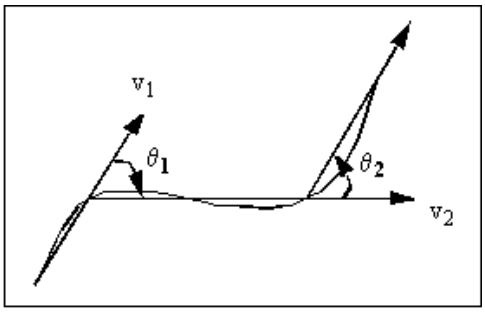


Figure 4. Relative lengths of segments and angles between them preserve local information, since they are invariant to similarity transformations.

In the speed curve y indicates object speed; a horizontal line describes an object moving at constant speed, while a line $y = ax + b$ describes an object with constant acceleration a. Rotation of the curve matters for speed curves so the angle of each feature value stands for the angle between the segment and the horizontal axis. Unlike Chen and Chang[12] we do not segment trajectories.

In Fig. 5 a longer curve C2 has segments similar to C1. We find their feature points and then calculate their feature values. At bottom the pattern of C1 also appears in C2 twice, and the corresponding feature values of C2 reflect the repeated pattern.

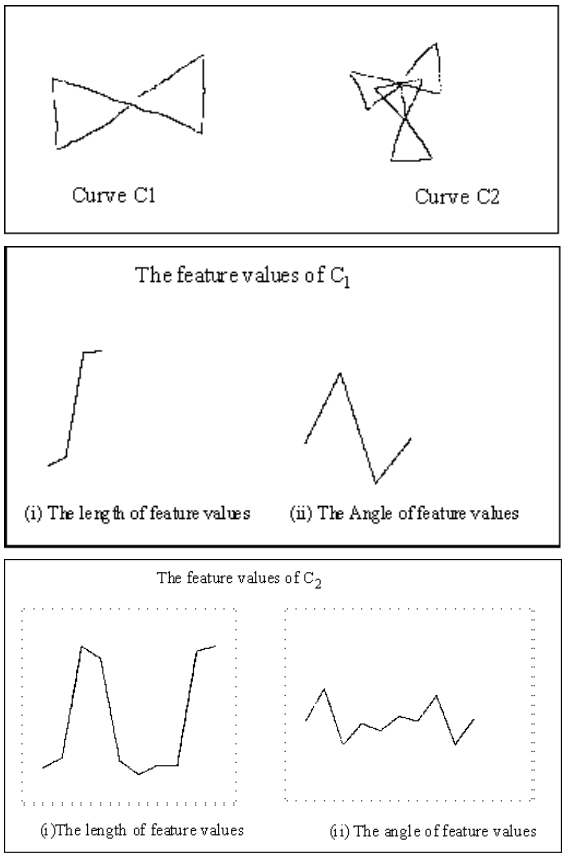


Figure 5. Curve C1 appears twice as a subsequence of C2, at the beginning of C2 and at its end.

We use a simple local comparison of feature values, the pointwise sum of the Euclidean distance. Other metrics may be more appropriate but this has been sufficient since the curves are already abstracted into features.

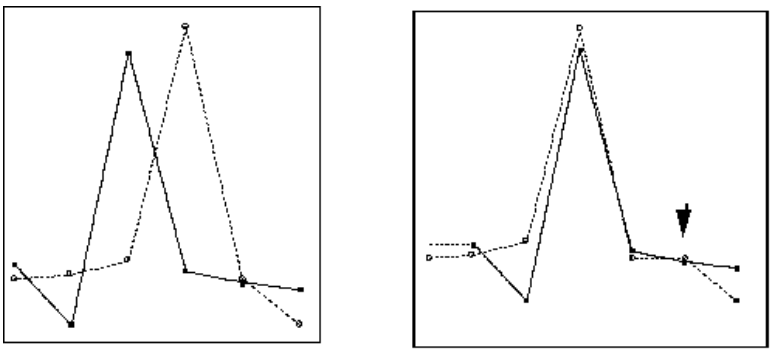


Figure 6. Warping: $V1$ (solid), $V2$ dashed. $V2$ has been shifted left and its second last point has been repeated.

2.2. Warping

Since feature points are decided by approximating maximum curvature, two curves with similar shapes may have a different number of feature points. Therefore, we use warping[16] when we measure the distance of the two linked lists. When we compare two sequences, points at different positions in the sequence may be closer to each other than the points at the same positions. Warping repeats and shifts elements so that the two sequences match better. Given a sequence V with N points in feature space, e.g., $(v_1, v_2, \ldots, v_n)$, let $Head(v)$ denote v_1 and $Rest(v)$ denote $(v_2, \ldots, v_n)$. We want to allow the stuttering transformation, i.e., repeating v_i and shifting elements. For sequences v and w, the warping distance is defined as follows:

$D_{warp}(nil, nil) = 0$
$D_{warp}(v, nil) = D_{warp}(w, nil) = \infty$
$D_{warp}(v, w) = D_{base}(Head(v), Head(w)) + min$ of
$\quad D_{warp}(v, Rest(w))$
$\quad D_{warp}(Rest(v), w)$
$\quad D_{warp}(Rest(v), Rest(w))$

nil denotes a null sequence; D_{base} is Euclidean distance.

Dynamic programming warps sequences of length n in $O(n)$ time. We test our algorithm with two angle vectors: $V1 = \{-42, -85, 105, -47, -54, -61\}$, $V2 = \{-54, -49, -40, 122, -54, -88\}$. After warping (Fig. 6), the point with the arrow has been repeated and the first portion of $V2$ has been shifted left.

Figure 7 shows the result of comparing a group of curves: we compare all the curves with the curve C1 – the number beside each curve indicates its distance from C1. The result resembles our perception of the curves quite well. C8 and C9 look most like C1 and our similarity metric ranks them best.

When we compare the speed curves, only those curves with the same shape without rotation should be considered similar. Among the curves in Fig. 8, C2 and C5 are both similar to C1, but C5 is closer to C1 than C2, so C5 has less distance than C2, and this is what we want.

3. FEATURE ORGANIZATION AND RETRIEVAL

We need a method of organizing the features for efficient search. R-trees[17] store the Minimum Bounding Rectangle (MBR), the smallest rectangle (perhaps multi-dimensional) that covers the geometric shapes. We chose R-Trees because they provide considerable performance advantages for spatial queries. Our indexing structure organizes a collection of tuples representing video sequences. Each tuple has a unique integer identifier. Leaf nodes contain index record entries of the form: (tuple-identifier, I, object-pointer), where tuple-identifier refers to a tuple in the database and I is the MBR of the feature vector. We use two R-trees to index the path and speed separately.

We provide a QBE (Query By Example) graphics interface. The query result is the set of sequences whose distance from the example query is less than a threshold T, up to a maximum number k. Query processing includes extracting the feature values of the query example, searching the index and presenting the results.

The similarity measurement on trajectories is computationally expensive. We propose a hierarchical searching strategy. First we search through the R-tree, comparing the MBR of the objects. If two objects are similar, their MBRs overlap in most cases. Conversely, if the MBRs of two objects do not overlap, they don't resemble each other.

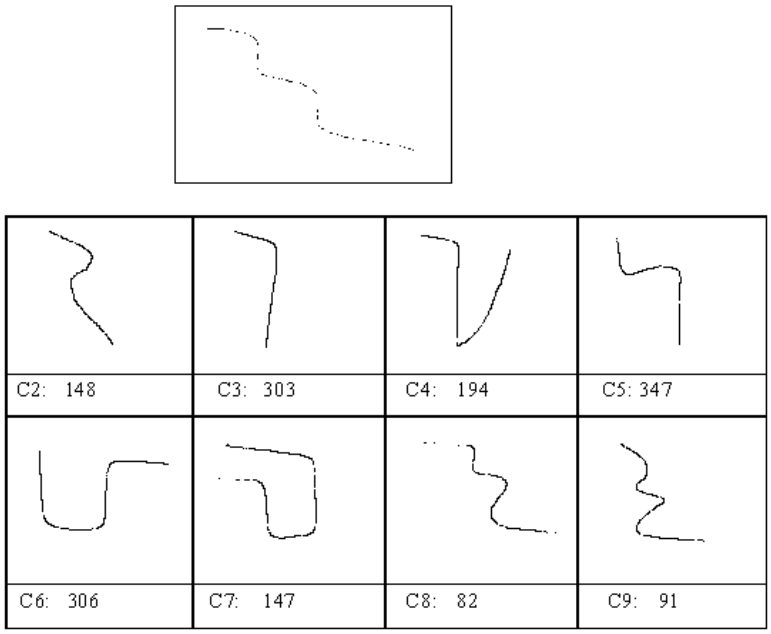


Figure 7. Path similarity: C8 and C9 are closest to the query curve shown above.

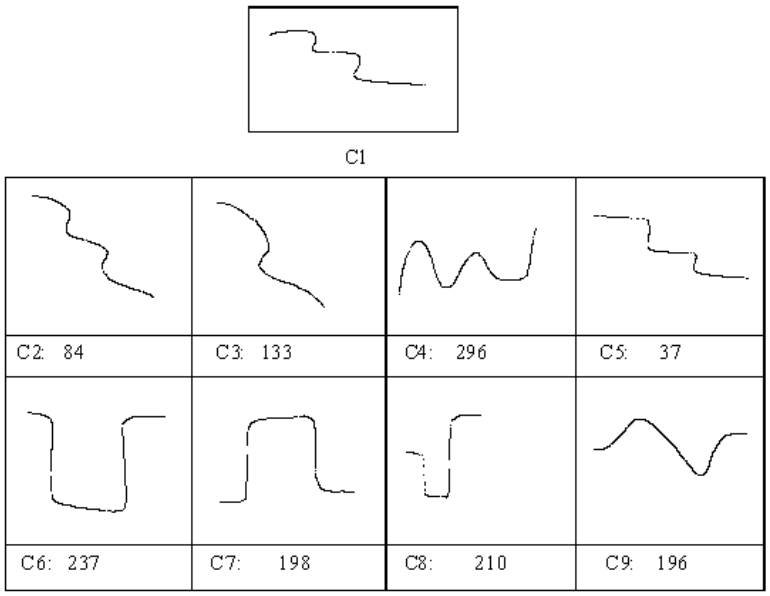


Figure 8. Speed similarity: C5 and C1 are closest to the query curve shown above.

Those objects whose MBRs do not overlap with the query are eliminated. At the second level, we check the angle attribute of feature values. We extract the angle dimension of each feature value, both from the query and the data in the database, and form vectors from them. Then we apply the 1D distance measurement metric with warping. Only those candidates whose distance is less than T are retained. Next, by checking the length we can get the third level candidates. The method we use is exactly the same as above, only now we extract the length parameter instead of angle. Most objects in the database are quickly eliminated from consideration by inexpensive computations in hierarchical searching. Only the remaining objects are filtered progressively by more expensive representations.

The query can be either based on path or on speed or on their combination. If the query is based on both speed and path, we execute the query procedure separately to get two result sets and then intersect them.

We implemented a video storage and retrieve system using Java. The system integrates curve feature extraction,

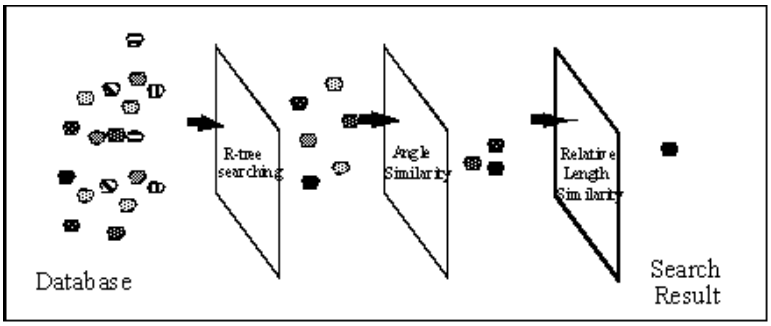


Figure 9. Hierarchical search

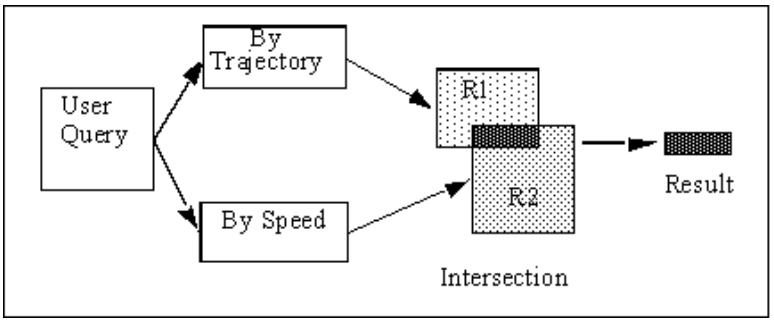


Figure 10. Query execution

similarity measurement and R-tree indexing. The user can track the position of an object in a video sequence. The speed curve can be automatically calculated and drawn. The user can also specify a path and a speed curve as a query. In Fig. 11 we set the threshold of similarity to 300. A query based on the path shown at bottom left retrieves the first row, that is ta1, ta2, ta3; and the query based on speed curve (bottom right) retrieves the middle column, that is ta2, tc2, tb2. The final result reports the intersection of them, which is ta2.

4. SUMMARY

We have identified several problems with previous work: features that are not invariant, inadequate indexing, and expensive comparison techniques. We proposed a novel representation of trajectories, *path* and *speed* curves, thus separating information about the path in the image, which may be well known, from the temporal information. We describe features of the curves that are invariant to scaling and rigid motions. Warping during matching permits flexible matching that is resistant to small variations. To examine whether the feature values can be indexed efficiently, we implemented an R-tree indexing structure for our features, and proposed a hierarchical search strategy. A QBE interface provides a framework for testing our ideas. Experiments show that the indexing structure works well with our feature vectors.

REFERENCES

1. A. Hampapur, R. Jain, and T. Weymouth, "Indexing in video databases," *SPIE* **2420**, pp. 292–306, 1995.
2. A. Pentland, R. Picard, and S. Sclaroff, "Photobook: Content-based manipulation of image databases," *IJCV* **18**, pp. 233–254, June 1996.
3. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image and video content: The qbic system," *Computer* **28**, pp. 23–32, September 1995.
4. A. Hamrapur, A. Gupta, B. Horowitz, C. Shu, C. Fuller, J. Bach, M. Gorkani, and R. Jain, "Virage video engine," in *SPIE Proceedings on Storage and Retrieval for Image and Video Databases V*, pp. 188–97, 1997.
5. J. Smith, *Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis.* PhD thesis, 1997.

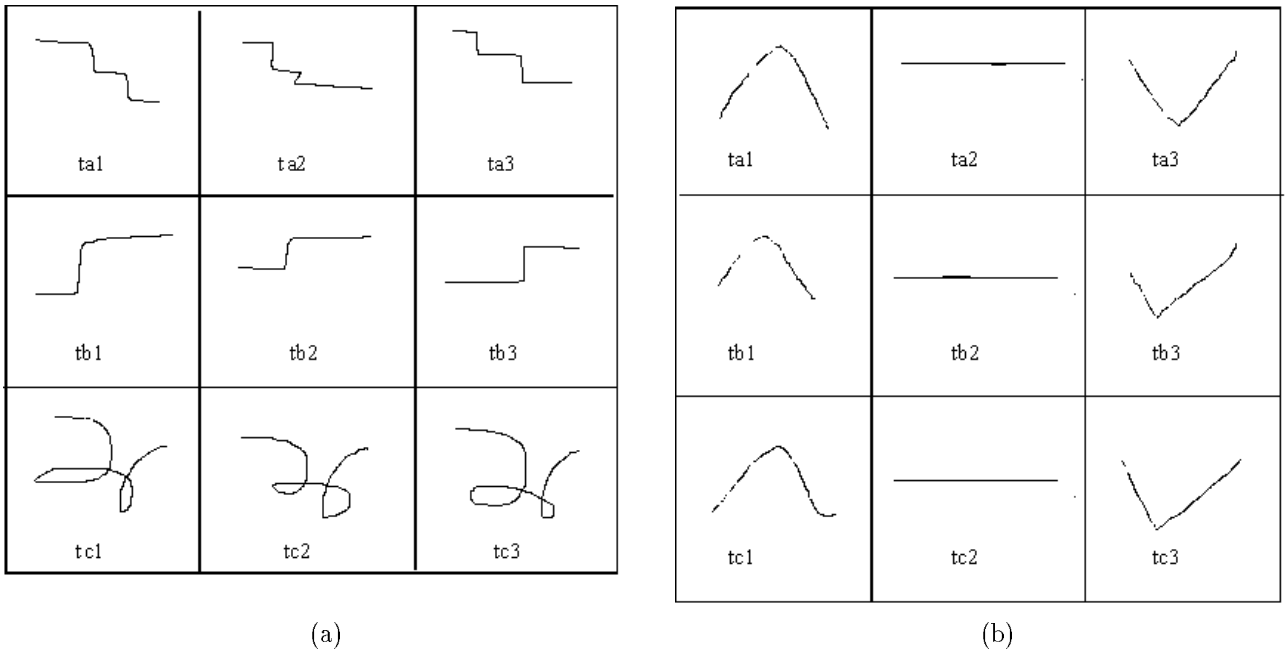


Figure 11. The paths (a) and speed curves (b) and query for our test example.

6. S.-F. Chang, W. Chen, H. Meng, H. Sundaram, and D. Zhong, "An automated content-based video search system using visual cues," in *ACM Multimedia 1997*, 1997.
7. M. Ioka and M. Kurokawa, "Estimation of motion vectors and their application to scene retrieval," *MVA* **7**(3), pp. 199–208, 1994.
8. E. Sahouria and A. Zakhor, "Motion indexing of video," in *ICIP97*, pp. II:526–xx, 1997.
9. S.-Y. Lee and H.-M. Kao, "Video indexing – an approach based on moving object and track," in *Proceedings of IS and T/SPIE Conference on Storage and Retrieval for Image and Video Databases I, Vol. SPIE 1908*, pp. 25–36, 1993.
10. Y. Deng and B. Manjunath, "Content-based search of video using color, texture, and motion," in *ICIP97*, pp. II:534–xx, 1997.
11. N. Dimitrova and F. Golshani, "Rx for semantic video database retrieval," in *ACM Multimedia Conference*, pp. 219–226, 1994.
12. W. Chen and S.-F. Chang, "Motion trajectory matching of video objects," in *SPIE Electronic Imaging*, Jan. 2000.
13. M. Shah, K. Rangarajan, and P. Tsai, "Motion trajectories," *SMC* **23**, pp. 1138–1150, 1993.
14. C. Chang, S. Hwang, and D. Buehrer, "A shape recognition scheme based on relative distances of feature points from the centroid," *PR* **24**, pp. 1053–1063, 1991.
15. F. Stein and G. Medioni, "Structural indexing: Efficient two dimensional object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**, pp. 1198–1204, Dec. 1992.
16. B. Yi, H. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," in *IEEE Conf. on Data Engineering (ICDE '98)*, pp. 300–400, 1998.
17. A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *Proc. ACM SIGMOD, 47*, 1984.